1. *Naive Bayes Gaussian discriminant analysis.* Consider the generative model for (supervised) classification by assuming $\Pr[y_i = c] = \pi_c$ and $\boldsymbol{x}_i | y_i \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$. Without any additional assumptions, this is the Gaussian discriminant analysis (aka quadratic discriminant analysis) that we covered in class. We've discussed one special case when all the $\boldsymbol{\Sigma}_c$ matrices are the same, which leads to the linear discriminant analysis (LDA) model.

   Here we make a different assumption: all the $\boldsymbol{\Sigma}_c$ matrices are diagonal (but not necessarily the same); for multivariate normals it means all variables are independent (conditioned on observing labels).

   (a) Derive the maximum likelihood estimate for the model parameters $\pi_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c, c = 1, \ldots, k$.

   (b) Given a new data point $\boldsymbol{x}_0$, explain how to predict $\widehat{y}_0$. Simplify the expression as much as possible.

Ans.1 =

Part(a) =

Bayes theorem explains that.=

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

Hence we can say that,

$$P(A|B) \propto P(B|A)\,P(A)$$

In this case $A \Rightarrow [y_i = c]$

and $B = x_i$

Then we can write that =

$$P(y_i = C \mid x_i) \propto P(x_i \mid y_i = C)\, P(y_i = C)$$

Here $P(y_i = C) = \pi$

And $P(x_i \mid y_i = C) = \mathcal{N}(\mu_C, \Sigma_C)$

Putting these values we get :

$$P(y_i = C \mid x_i) = \pi_C \, \mathcal{N}(\mu_C, \Sigma_C)$$

$$\Rightarrow \pi_C \det(2\pi\Sigma_C)^{-1/2} \exp\left(-\frac{1}{2}(x_i - \mu_C)^T \Sigma_C^{-1} (x_i - \mu_C)\right)$$

Maximum log likelihood function becomes

$$\Rightarrow \sum_{i=1}^{n} \log \pi_C - \frac{1}{2} \log 2\pi \det \Sigma_C - \frac{1}{2}(x_i - \mu_C)^T \Sigma_C^{-1} (x_i - \mu_C)$$

(1) = Taking derivative of MLE function with respect to π_C

$$MLE = \sum_{i=1}^{n} \log \pi_C \quad \text{(ignoring all constants)}$$

construct lagrangian with λ =

$$L(\pi_c) = \sum_{i=1}^{n} \log \pi_c + \lambda \pi_c$$

$$\Rightarrow \frac{\partial L(\pi_c)}{\partial \pi_c} = \sum_{i=1}^{n} \frac{1}{\pi_c} + \lambda \cdot 1$$

(equate to 0)

$$\Rightarrow \sum_{i=1}^{n} \frac{1}{\pi_c} + \lambda = 0$$

$$\Rightarrow \pi_c = \frac{-n_c}{\lambda}$$

taking summation -

$$\sum_{c=1}^{N} \pi_c = \frac{-\sum_{c=1}^{N} (n_c)}{\lambda}$$

$$\boxed{\lambda = \frac{-N_c}{N}}$$

$$\pi_c = \frac{\text{Total no. of samples in class } C}{\text{Total no. of samples in all classes}}$$

(2) = calculating $\Sigma_c \Rightarrow$

Partial derivative of MLE function w.r.t. Σ_c we obtain =

$$\sum_{i=1}^{N} \left(\frac{\partial}{\partial \Sigma_c} \left(-\frac{1}{2} \log 2\pi \det \Sigma_c \right) - \frac{\partial}{\partial \Sigma_c} \left[(x_i - \mu_c)^T \Sigma_c^{-1} (x_i - \mu_c) \right] \right)$$

Solving it we get =

$$\sum_{i=1}^{n} \left[\frac{-1}{2} \Sigma_c^{-1} + \frac{1}{2} \Sigma_c^{-1} (x_i - \mu_c)(x_i - \mu_c)^T \Sigma_c^{-1} \right] = 0$$

$$\sum_{i=1}^{n} \left[\frac{-1}{2} \left(I - \Sigma_c^{-1} (x_i - \mu_c)(x_i - \mu_c)^T \right) \Sigma_c^{-1} \right] = 0$$

multiplying both sides by $\Sigma_c \Rightarrow$

$$\sum_{i=1}^{n} \Sigma_c = \sum_{i=1}^{n} (x_i - \mu_c)(x_i - \mu_c)^T$$

Then Σ_c can be given as =

$$\Sigma_C = \frac{1}{N_C} \sum_{C=1}^{N} (x_i - \mu_C)(x_i - \mu_C)^T$$

③ = Obtaining μ_C.

Taking derivative of MLE w.r.t. μ_C.

$$\frac{\partial}{\partial \mu_C} \left[\frac{-1}{2} (x_i - \mu_C)^T \Sigma_C^{-1} (x_i - \mu_C) \right]$$

remaining all constant terms become = 0.

Σ_C^{-1} = diagonal ⇒ mean symmetric

for Symmetric matrices =

$$\frac{\partial}{\partial A} A^T B A = 2BA$$

Hence above equation simplifies to =

$$\sum_{i=1}^{n} \Sigma_C^{-1} (x_i - \mu_C) = 0$$

which gives us:

$$\mu_c = \frac{1}{N_c} \sum_{i=1}^{N} x_i$$

Part b ⇒ Prediction $(\hat{y}_0)$ for given value of x_0.

$$\hat{y}_0 = f(x_0) = \underset{c}{\operatorname{argmax}}\ P(y_i = c) \cdot P(x | y = y$$

$$= \underset{c}{\operatorname{argmax}}\ \pi_c \cdot N(\mu_c, \Sigma_c)$$

$$= \underset{c}{\operatorname{argmax}} \left(\log \pi_c - \frac{1}{2} (x - \mu_c)^T \Sigma_c^{-1} (x - \mu_c) \right)$$

remove all constant term

$$= \underset{c}{\operatorname{argmax}} \left[-\frac{1}{2} x^T \Sigma_c^{-1} x + x^T \Sigma_c^{-1} \mu_k - \frac{1}{2} \mu_c^T \Sigma_c^{-1} \mu_c + \log \pi_c \right]$$

2. Consider the lasso regularized k-class logistic regression problem

$$\underset{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_k}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n}\left(\log\sum_{c=1}^{k}\exp(\boldsymbol{\theta}_c^\top\boldsymbol{\phi}_i)-\boldsymbol{\theta}_{y_i}^\top\boldsymbol{\phi}_i\right)+\frac{\lambda}{2}\sum_{c=1}^{k}\|\boldsymbol{\theta}_c\|_1.$$

Write the pseudocode of the proximal stochastic gradient descent algorithm for solving it.

Hint: Denote $f(\boldsymbol{z})=\log\sum\exp(\boldsymbol{z})$, then $\nabla f(\boldsymbol{z})=\frac{1}{\sum\exp(\boldsymbol{z})}\exp(\boldsymbol{z})$, where we overload the definition of $\exp(\cdot)$ for vector inputs by taking exponential element-wise. If $g(\boldsymbol{x})=f(\boldsymbol{A}\boldsymbol{x})$, then $\nabla g(\boldsymbol{x})=\boldsymbol{A}^\top\nabla f(\boldsymbol{A}\boldsymbol{x})$.

(a) = In order to minimize the objective function of the form ⇒

$$\underset{\theta}{\text{minimize}} \quad L(\theta)+\lambda r(\theta)$$

Here $L(\theta)=\frac{1}{n}\sum_{i=1}^{n}\left(\log\sum_{c=1}^{K}\exp(\theta_c^T\phi_i)-\theta_{y_i}^T\phi_i\right)$

$L(\theta)$ = Loss function

and $r(\theta)$ = Regulizer

$$r(\theta)=\sum_{c=1}^{K}\|\theta_c\|_1$$

In order to minimize this function and obtain the optimized values of Parameters θ we have to use Proximal gradient descent.

Proximal gradient descent is given as =

$$\theta^{(t+1)} = \text{Prox}_{\gamma\lambda r}(\theta^t - \gamma \nabla L(\theta^t))$$

$$= \underset{\theta}{\text{argmin}} \frac{1}{2} \|\theta - \theta^t + \gamma \nabla L(\theta^t)\|^2 + \lambda \gamma r(\theta)$$

In order to obtain Proximal gradient we have to obtain gradient of loss function first. i.e. $L(\theta)$

Pseudocode for calculating Proximal gradient descent =

① = initially assume parameter (θ) values.

② = obtain gradient of loss function $L(\theta)$.

Let's derive the gradient of $L(\theta)$.

$$L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left(\log \sum_{c=1}^{K} \exp(\theta_c^T \phi_i) - \theta_{y_i}^T \phi_i \right)$$

We can assume that =

$$f(z) = \log \sum \exp(z), \quad z = \theta_c^T \phi_i$$

Then we can write loss function as =

$$L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left(f(z) - \theta_{y_i}^T \phi_i \right)$$

Taking gradient =

$$\nabla L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left(\nabla f(z) - (\nabla \theta_{y_i}^T) \cdot \phi_i - (\nabla \phi_i) \theta_{y_i}^T \right)$$

We are given that

$$\nabla f(z) = \frac{1}{\sum \exp(z)} \exp(z)$$

Then we can write ⇒

$$\nabla L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left[\frac{1}{\sum \exp(z)} \exp(z) - \nabla(\theta_{y_i}^T) \phi_i - (\nabla \phi_i) \theta_{y_i}^T \right]$$

Putting value of $z = \theta_c^T \phi_i$

We can write the loss function as =

$$\nabla L(\theta) = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{\sum_{c=1}^{K}\exp(\theta_c^T\phi_i)}\exp(\theta_c^T\phi_i)(\nabla\theta_{y_i}^T)\phi_i - (\nabla\phi_i)\theta_{y_i}^T\right]$$

(3) = Put the assumed value of parameters θ and value of one random sample from the data set (ϕ) in this loss function.

(4) = Calculate value of $\theta^{(t+1)}$ for current iteration using proximal mapping formula.

$$\theta^{(t+1)} = \text{Prox}_{\gamma\lambda r}(\theta^t - \gamma\nabla L(\theta^t))$$

$$= \underset{\theta}{\text{argmin}}\ \frac{1}{2}\|\theta - \theta^t + \gamma\nabla L(\theta^t)\|^2 + \lambda\gamma r(\theta)$$

(5) = These steps will be iterated untill the value of θ parameter converge.

3. Consider the latent variable model with the following probability distribution:

$$\Pr(y_i = c) = \pi_c, \quad \Pr(\boldsymbol{x}_i | y_i = c) \sim \text{Multi}(\boldsymbol{p}_c, L_i),$$

meaning that y_i is categorical, with k possible outcomes, and $\Pr(y_i = c) = \pi_c$; $(\boldsymbol{x}_i | y_i = c)$ follows a multinomial distribution by drawing from $\boldsymbol{p}_c$ L_i times, i.e.,

$$p(\boldsymbol{x}_i | y_i = c) = \frac{L_i!}{\prod_{j=1}^{d} x_{ij}!} \prod_{j=1}^{d} {p_{cj}}^{x_{ij}}.$$

Given data samples $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$:

(a) Write out the maximum likelihood formulation for estimating $\boldsymbol{p}_1, ..., \boldsymbol{p}_k$, and $\boldsymbol{\pi}$. Simplify the objective function as much as possible.

(b) Derive an expectation-maximization algorithm for approximately solving the aforementioned problem.

(c) Implement this algorithm and try it on the 20 News Group data set with $k = 20$ (on the raw word-count data, without tf-idf preprocessing). Show the top 10 words in each cluster.

Ans.3 =

Part(a) =

given that $Pr(y_i = c) = \pi_c$

and $Pr(x_i | y_i = c) = \text{Multi}(p_c, L_i)$

also given that

$$P(x_i | y_i = c) = \frac{L_i!}{\prod_{j=1}^{d} x_{ij}!} \prod_{j=1}^{d} {p_{cj}}^{x_{ij}}$$

let θ be $(\rho_1, \rho_2 \dots \pi)$ is our shorthand unknown parameter.

max. log likelihood =

$$l_c(\theta; y) = \sum_{i=1}^{n} \log p(x_i, y_i; \theta)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{k} \pi(y_i = j) \log (\pi_j \phi(x_i; \rho_j))$$

Part (b) = algorithm =

① = initialize $\theta^{(1)}$ arbitrarily.

② = alternate b/w E & M step until convergence.

a) = E-step = $q^{(t+1)} = \text{argmax}_q L(q, \theta^t)$

b) = M-step = $\theta^{(t+1)} = \text{argmax}_\theta L(q^{t+1}, \theta)$

where $L(q, \theta)$ = Lower bound/by term auxillary function

The solution is,

$$q^{(t+1)}(y) = p(y|x;\theta^t)$$

4. *Multidimensional scaling (MDS).* MDS is another classical approach for unsupervised embedding, and to some extent relates to PCA.

   The main idea of MDS is to embed each $\boldsymbol{x}_i$ to $\boldsymbol{y}_i$ so that the pair-wise distances are preserved as much as possible. This can be formulated as the following optimization problem

$$\underset{\boldsymbol{y}_1,\dots,\boldsymbol{y}_n}{\text{minimize}} \quad \sum_{i=1}^{n}\sum_{j=1}^{i-1}(\|\boldsymbol{x}_i-\boldsymbol{x}_j\|-\|\boldsymbol{y}_i-\boldsymbol{y}_j\|)^2.$$

   There is no close-form solution for this formulation. A modified formulation called *classical scaling* is proposed:

$$\underset{\boldsymbol{y}_1,\dots,\boldsymbol{y}_n}{\text{minimize}} \quad \sum_{i=1}^{n}\sum_{j=1}^{n}(\widetilde{\boldsymbol{x}}_i^\top\widetilde{\boldsymbol{x}}_j-\boldsymbol{y}_i^\top\boldsymbol{y}_j)^2, \tag{1}$$

   where $\widetilde{\boldsymbol{x}}_i=\boldsymbol{x}_i-(1/n)\sum_{j=1}^{n}\boldsymbol{x}_j$ is the centered data.

   (a) Use the Eckart-Young theorem to show that an optimal solution of (1) is to take the eigen-decomposition of the matrix $\widetilde{\boldsymbol{X}}^\top\widetilde{\boldsymbol{X}}=\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$, where $\widetilde{\boldsymbol{X}}=[\ \widetilde{\boldsymbol{x}}_1\ \cdots\ \widetilde{\boldsymbol{x}}_n\ ]$, keep the k largest eigenvalues in $\boldsymbol{\Lambda}$ and the corresponding columns in $\boldsymbol{U}$, and let $\boldsymbol{Y}=\boldsymbol{\Lambda}^{1/2}\boldsymbol{U}^\top$; then each $\boldsymbol{y}_i$ is the ith column of $\boldsymbol{Y}$.

   (b) Oftentimes one is directly given the pair-wise distance matrix $\boldsymbol{D}$, where

$$D_{ij}=\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2,$$

   without explicitly given the data points $\boldsymbol{x}_1,\dots,\boldsymbol{x}_n$. Show that we can define the matrix

$$\boldsymbol{B}=\left(\boldsymbol{I}-\frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top\right)\boldsymbol{D}\left(\boldsymbol{I}-\frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top\right),$$

   and replace $\widetilde{\boldsymbol{x}}_i^\top\widetilde{\boldsymbol{x}}_j$ with B_{ij} in formulation (1).

Part (a) =

formulated Problem for optimization =

$$\underset{y_1,y_2,\dots,y_n}{\text{minimize}} \quad \sum_{i=1}^{n}\sum_{j=1}^{i-1}\left(\|x_i-x_j\|-\|y_i-y_j\|\right)^2$$

Clamical Scaling - modified solution is -

$$\underset{y_1, y_2 - y_n}{\text{minimize}} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\tilde{x}^T \tilde{x}_j - y_i^T y_j \right)^2$$

writing this equation in Matrix form -

$$\underset{Y}{\text{minimize}} \left\| \tilde{X}^T \tilde{X} - Y^T Y \right\|$$

Eckart-young theorem giver a low rank approximation, it giver solution to solve minimization Problems. The cost function evaluates the fit between the data matrix and an approximating matrix which has reduced rank.

mathematically Eckart-Young theorem can be written as -

for given matrices P and Q $\Rightarrow$

$$\| P - Q \| \geq \| P - P_r \|$$

where (rank of matrix Q) $\leq r$

also P_r represents the best approximation of P for rank r, conniderng the largest k-eigen valuer and correspondi -ng columns in SVD.

using the Eckart-young theorem for the above poblen we can write=

$$\| \tilde{X}^T \tilde{X} - Y^T Y \| \geq \| \tilde{X}^T \tilde{X} - (\tilde{X}^T \tilde{X})_k \|$$

where we can say =

$(\tilde{X}^T \tilde{X})_k$ = approximation matrix with rank k.

and rank of matrix $(Y^T Y) \leq k$.

Here we want to enbed input vector into a reduced dimension k.

we can say at optimal point =

$$\boxed{Y^T Y = \left(\tilde{X}^T \tilde{X}\right)_k} \longrightarrow \text{eqtn } ①$$

we can write SVD for $\tilde{X}^T X$ as =

$$\tilde{X}^T \tilde{X} = U \Lambda U^T$$

$\tilde{X}^T \tilde{X}$ is symmetric and Hence taking the largest K eigen values and corresponding columns in U we can write above equation as =

$$\left(\tilde{X}^T \tilde{X}\right)_k = U_k \Lambda_k U_k^T$$

Putting the value of $\left(\tilde{X}^T \tilde{X}\right)_k$ from equation ① for point of optimality we can write =

$$Y^T Y = U_k \Lambda_k U_k^T$$

Here $Y^T Y$ - represents a Positive Definite value.

if $Y^T Y \in$ (+ve, Definite value)

Then $\boxed{Y = \Lambda_k^{1/2} U_k^T}$

where $\left(\Lambda_k^{\frac{1}{2}}\right)$ is also a positive non negative value.

we have seen that $\left(Y = \Lambda_k^{1/2} U_k^T\right)$ after eigen value decomposition of $\tilde{X}^T \tilde{X}$. we have got only k largest eigen values in Λ and corresponding columns in U.

Hence Y could be evaluated using relationship $\boxed{Y = \Lambda^{1/2} U^T}$ Proved

Part (b) =

The general conceptual Process behind the idea of MDS is that →

(1) = First it transforms the squared distance matrix D to an inner product form.

(2) = Secondly it computes the eigen decomposition for this inner product form.

We are given the distance matrix D. for which

$$D_{ij} = \| x_i - x_j \|^2$$

$$D_{ij} = (x_i - x_j)^T (x_i - x_j)$$

$$D_{ij} = x_i^T x_i + x_j^T x_j - 2 x_i^T x_j$$

→ equation (1)

lets apply the above mentioned process on given distance matrix $D =$

lets assume that K is the inner product matrix of X.

$$K = X^T X \longrightarrow \text{equation (2)}$$

with $K = diag(K_{ii}) \in R^n$, Hence we can write equation (1) by putting values of $X^T X$ from equation (2) as

$$D = (d_{ij}^2) = K \cdot 1^T + 1 \cdot K^T - 2K$$

also we know that =

$$\tilde{x}_i = x_i - mean(X) = x_i - \frac{1}{n} X \cdot 1$$

$$\Rightarrow \tilde{X} = X - \frac{1}{n} X \cdot 1 \cdot 1^T \longrightarrow \text{eqtn (3)}$$

Then

$$\tilde{K} = \tilde{X}^T \tilde{X}$$

Putting value of $\tilde{X}$ from equation (3) we get =

$$\tilde{K} = \tilde{X}^T \tilde{X}$$

$$= \left(X - \frac{1}{n} X \cdot 1 \cdot 1^T\right)^T \left(X - \frac{1}{n} X \cdot 1 \cdot 1^T\right)$$

$$= K - \frac{1}{n} K \cdot 1 \cdot 1^T - \frac{1}{n} 1 \cdot 1^T \cdot K + \frac{1}{n^2} \cdot 1 \cdot 1^T \cdot K \cdot 1 \cdot 1^T$$

Let's assume that =

$$B = -\frac{1}{2} H \cdot D \cdot H^T$$

where $H = I - \frac{1}{n} 1 \cdot 1^T$

Here H is called the centering matrix.

Then $B = -\frac{1}{2} H \left(K \cdot 1^T + 1 \cdot K^T - 2K\right) \cdot H^T$

Since,

$$K \cdot 1^T \cdot H^T = K \cdot 1 \left(I - \frac{1}{n} 1 \cdot 1^T \right)$$

$$= K \cdot 1 - K \left(\frac{1^T \cdot 1}{n} \right) \cdot 1$$

$$= 0$$

Hence, $H \cdot K 1 \cdot H^T = 0$, and

$$H \cdot 1 K^T \cdot H^T = 0$$

Therefore,

$$B = H \cdot K \cdot H^T$$

$$= \left(I - \frac{1}{n} \cdot 1 \cdot 1^T \right) \cdot K \cdot \left(I - \frac{1}{n} \cdot 1 \cdot 1^T \right)$$

$$= K - \frac{1}{n} 1 1 \cdot K - \frac{1}{n} \cdot K \cdot 1 \cdot 1^T + \frac{1}{n^2} \cdot 1 \left(1^T \cdot K \cdot 1 \right) 1^T$$

$$= \tilde{K}$$

Hence $B = \tilde{K}$

and $\tilde{K} = \tilde{X}^T \tilde{X}$

we can write =

$$\boxed{B = \tilde{X}^T \tilde{X}}$$

Hence we have proved that given a distance matrix $D = (d_{ij}^2)$ we can convert it to an inner product matrix -

$$B = -\frac{1}{2} H D H^T$$

where $H = \left(I - \frac{1}{n} 1 \cdot 1^T\right)$

is called centering matrix.

Problem formulation by replacing

$$B_{ij} = \tilde{x}_i^T \tilde{x}_j$$

Hence,

$$\underset{y_1, y_2 \dots y_n}{\text{minimize}} \sum_{i=1}^{n} \sum_{j=1}^{n} (B_{ij} - y_i^T y_j)^2$$